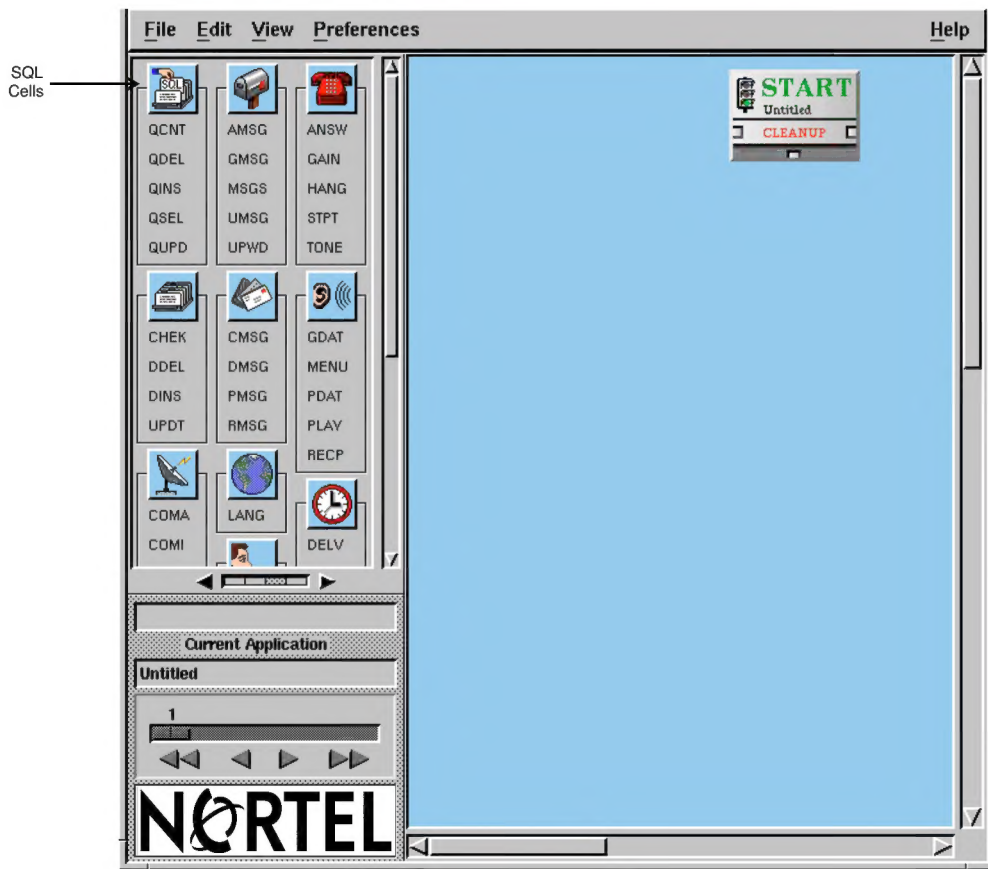


Chapter 2: Building an SQL application

The IVR 2.0/I SQL Server has five cells that you can include in your application to allow callers to manipulate data in your database. Figure 2-1 illustrates the different SQL server cell types as they appear on the Application Editor palette. This chapter describes each of the SQL cells that you can use in addition to the following information:

- Counting selected rows with QCNT (SQL SELECT COUNT)
- Deleting a row with QDEL (SQL DELETE)
- Inserting a new entry with QINS (SQL INSERT)
- Retrieving information with QSEL (SQL SELECT)
- Updating a row with QUPD (SQL Update)

Figure 2-1
The Application Editor palette with SQL cells



The SQL Server cells are designed for your convenience. If you have experience using standard and embedded SQL statements, you can use the SQL Server cells to quickly design IVR 2.0/I applications for accessing your database.

Note: SQL cells work within the limitations specified in the next section.

Common characteristics of SQL server cells

As shown in Figure 2-1, each of the SQL Server cells is represented in the Application Editor as a cell icon. You can place a cell in your application by selecting the appropriate cell type from the palette.

If an application uses SQL cells, specify the DBMS type, the database name, the server count, the user name and the password in the default cell as shown in Figure 2-2.

Each SQL cell includes:

- The table with the data which will be manipulated, as shown in Figure 2-3.
- The columns in those cells that manipulate column data and the associated value or buffer containing the value (that is, inserting, updating, or deleting values), as shown in Figure 2-3.
- The where clause, in those cells requiring selection criteria, as shown in Figure 2-3.

Figure 2-2
Default cell parameter

The screenshot shows a dialog box titled "DFLT Parameters". It contains several configuration fields:

- Maximum Pause Time (seconds):** A numeric field with a value of 15.
- Pause Timeout Action:** A dropdown menu set to "ABORT".
- Interdigit Timeout (seconds):** A numeric field with a value of 5.
- Minimum Voice Duration (milliseconds):** A numeric field with a value of 200.
- SQL DBMS Type:** A dropdown menu set to "<NONE>".
- SQL Database Name:** A dropdown menu set to "<NONE>".
- SQL Max Server Count:** A numeric field with a value of 1.
- User Name:** A text input field.
- Password:** A text input field with masked characters (asterisks).
- Host Name:** A dropdown menu set to "<NONE>".

At the bottom of the dialog are three buttons: "Apply", "Cancel", and "Help".

To the right of the dialog, a text box states: "The START Cell Parameter Page allows you to specify:" followed by a bulleted list:

- SQL DBMS Type
- Database Name
- Up to 5 Servers
- User Name and Password
- Host Name

SQL default cell parameters

You must specify the following SQL parameters in the Default cell:

SQL DBMS Type Specify the DBMS type that you are using for your application. Current DBMS types supported by IVR 2.0/I are: Informix, Ingres, Sybase, and Oracle.

SQL Database Name Specify the database name. The database must be previously set up. Although the buffer allows this name to be up to 31 characters long Informix only allows 8 characters, Sybase allows 12.

SQL Max Server Count Specify the number of servers to use (up to five). The number of servers to be specified depends on the number of queued requests. The performance of your SQL queries will suffer if you do not have an adequate number of SQL servers selected. The recommended number at this time is one.

Host Name Specify the remote host name. Use the host name only if the database is remote.

Figure 2-3
QSEL parameter

Specify the table, the column, and the WHERE clause in the appropriate fields. You can get a popup containing valid values for the where clause fields by clicking the right mouse button on the field. See parenthesis popup below.

Cell #4

QSEL SQL Select

Retrieve Row

Comments

SQL Table Name "order" ...

Column and Buffer Table

Column	Buffer
price	DATA EXCHANGE #1
	DATA EXCHANGE #1
	DATA EXCHANGE #1
	DATA EXCHANGE #1

Where Clause

Logical	Column	Type	Operator	Value	)'s
AND	order	NUMERIC	>=	1	
	(	STRING	!=	"pending"	
	(	STRING			
	((	STRING			

Apply Cancel Help

Specifying clauses

You can enter one or more clauses. Move the cursor to the field where you wish to enter a value and type a value. You can also select a value by selecting the field and pressing the right mouse button. A pop-up appears containing the valid values for that field. Move the cursor to the value you want and release the right mouse button. The value appears in the field. The possible fields where you can enter a value, as shown in Figure 2-3, are:

LOGICAL Enter a logical (Boolean) operator (**AND**, **NOT**, **ANDNOT**, **ORNOT**, or **OR**) when specifying a compound expression.

(

Enter left parenthesis as single, double, triple [(, ((, or (((].

COLUMN

Enter the name of a column, up to 31 characters.

Type

Press the button and choose the data type. Valid choices are:

- string
- numeric
- money
- date
- float

OPERATOR

Enter the relational operator, such as =, !=, >, <, >=, <= (equal, not equal, greater than, less than, greater than or equal to, less than or equal to).

VALUE

Enter a value against which the query is matched. The value can be a system buffer, an application buffer, a number, or a hard coded value surrounded by double quotes.

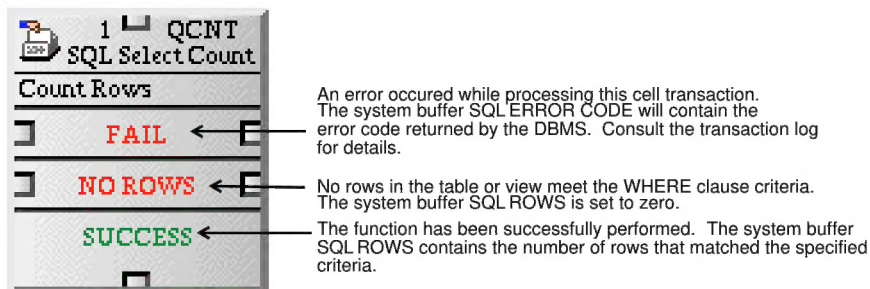
)

Enter closing parenthesis as single, double, or triple [),), or])].

Counting selected rows with QCNT (SQL select count)

Use the QCNT cell to count the number of rows in a table that match specific criteria. The QCNT cell parameters identify the table to be accessed and the WHERE clause selection criteria. Figure 2-4 shows the QCNT cell and the possible next cells. Figure 2-5 shows the QCNT parameters.

Figure 2-4
QCNT cell



Entering an SQL statement in the QCNT parameter

Suppose that you want to count all the rows in table “product” that have the value in the “size” column equal to “small.” Normally you would code the query to the database table by entering the following SQL statement:

```
SELECT COUNT (*) FROM product

WHERE size = 'small'
```

When you create the QCNT cell, you specify the following values in the cell parameters window as shown in Figure 2-5:

Screen location	Field	Specify
	SQL Table Name	product
Under the Where Clause	Column	size
	Operator	=
	Value (	“small”

Figure 2-5
QCNT parameters

Deleting a row with QDEL (SQL delete)

Use the QDEL cell to delete a row in a database table. The QDEL parameter identifies the table to be accessed. The rows to be deleted are specified by a WHERE clause. Figure 2-6 shows the QDEL cell and the possible next cells.

Figure 2-6
QDEL cell

An error occurred while processing this cell transaction. The system buffer SQL ERROR CODE will contain the error code returned by the DBMS.

No rows in the table or view meet the WHERE clause criteria. The system buffer SQL ROWS is set to zero.

The delete rows transaction was successful. The system buffer SQL ROWS will contain the number of rows that have been deleted.

Entering an SQL statement in a QDEL parameter

Suppose that you want to enable a caller to delete an order that is currently pending in the database. You would code the following SQL statement to delete the order:

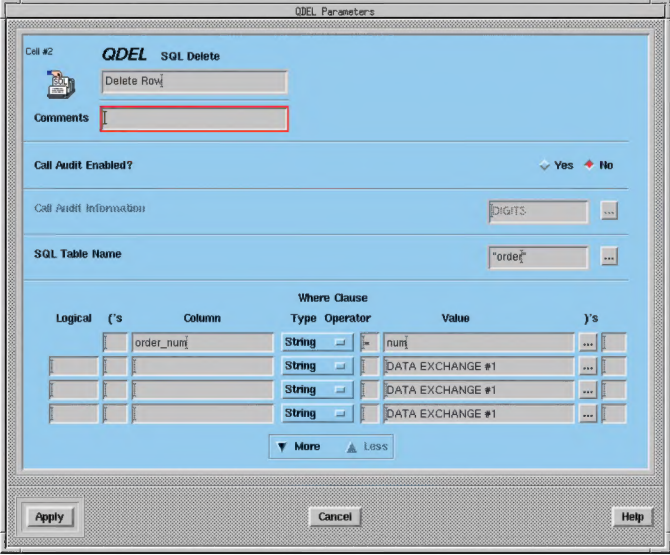
```
DELETE FROM order WHERE order_num = :num;
```

Note: Due to the syntax convention for the SQL DELETE statement, it is possible to delete all rows from a table using the statement: **DELETE from ORDER**. The Meridian IVR SQL cell performs this same statement if the WHERE clause is omitted.

When you create the QDEL cell, you specify the following values in the cell parameters window as shown in Figure 2-7

Screen Location	Field	Specify
	SQL Table Name	order
Under the Where Clause	Column	order_num
	Operator	=
	Value (	num

Figure 2-7
QDEL parameter



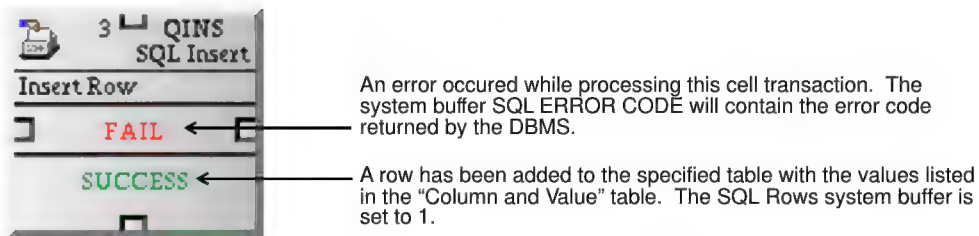
The image shows a 'QDEL Parameters' dialog box. At the top, it says 'Cell #2' and 'QDEL SQL Delete'. Below this is a 'Delete Row' button and a 'Comments' text field. A section for 'Call Audit Enabled?' has 'Yes' and 'No' radio buttons, with 'No' selected. Below that is 'Cell Audit Information' with a 'DIGITS' field and a '...' button. The 'SQL Table Name' field contains 'order' and also has a '...' button. The main section is titled 'Where Clause' and contains a table with columns: Logical, ('s, Column, Type, Operator, Value, and)'s. The table has four rows, all with 'String' type and '=' operator. The values are 'num', 'DATA EXCHANGE #1', 'DATA EXCHANGE #1', and 'DATA EXCHANGE #1'. Below the table are 'More' and 'Less' buttons. At the bottom of the dialog are 'Apply', 'Cancel', and 'Help' buttons.

Logical	('s	Column	Type	Operator	Value	)'s
		order_num	String	=	num	
			String	=	DATA EXCHANGE #1	
			String	=	DATA EXCHANGE #1	
			String	=	DATA EXCHANGE #1	

Inserting a row with QINS (SQL insert)

Use the QINS cell to insert a row of data into a database table. The QINS parameter identifies the table to be accessed. The Column and Value table specifies each column in a new row, the column type, and the corresponding value to be inserted. The column “type” can be numeric, string, money, date, or float. The “value” you specify can be a specific value or a buffer containing the value. Figure 2-8 shows the QINS cell and the possible next cells.

Figure 2-8
QINS cell



Entering an SQL statement into a QINS parameter

Suppose that you want to enter the name, company, two separate addresses, city, state, zip code, and phone number of a customer into a specific database. You would identify the table and columns that would be affected and the corresponding buffers with the following SQL statement:

```
INSERT INTO cust (name, company, address1, address2, city,
state, zip, phone)

VALUES (:NAME, :COMPANY, :BUFFER1, :BUFFER2, :BUFFER3,

:BUFFER4, :ZIP, :DIGITS);
```

When you create the QINS cell, you specify the following values in the cell parameters window as shown in Figure 2-9. Note that the “TYPE” changes to “numeric” for the zip value and the phone value.

Figure 2-9
QINS parameter

Cell #3 **QINS SQL Insert**

Insert Row

Comments

Call Audit Enabled? ☒ Yes ☐ No

Call Audit Information GIFS

SQL Table Name "CUST"

Column	Type	Value
name	String	NAME
company	String	COMPANY
address1	String	BUFFER1
address2	String	BUFFER2

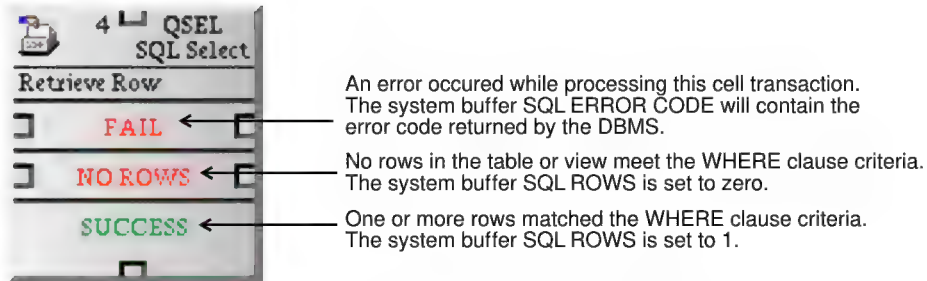
More Less

Apply Cancel Help

Retrieving information with QSEL (SQL select)

Use the QSEL cell to retrieve a single row of data from a table. The QSEL parameters identify the table or view to be accessed; the columns whose values should be returned; and the WHERE clause selection criteria. You can specify the next cell based on the number of rows selected, whether an error occurs, or if a row is successfully returned. Figure 2-10 shows a QSEL cell and the possible next cells.

Figure 2-10
QSEL cell



Note: Unlike a SELECT statement that can retrieve numerous rows, QSEL only returns one row.

Entering an SQL statement into a QSEL parameter

Suppose that you want to retrieve a row where “order” is greater than or equal to “1” and where “status” is not “pending.” You would code the SQL statement to retrieve a row meeting these characteristics:

- SELECT PRICE FROM order
- WHERE num >= 1
- AND status != pending

Use the values described in Table 2-1. They specify what should be returned when a row is retrieved as shown in Figure 2-11.

Table 2-1
Coding an SQL statement

Screen Location	Field	Specify
	SQL Table Name	order
Under the Column and Buffer Table	Column	price
Under the Where Clause	Column	num
	Operator	>=
	Value (	1
For the "And" clause	Column	status
	Operator	!=
	Value (	pending

Figure 2-11
QSEL parameters

QSEL Parameters

Cell #3

QSEL SQL Select

Retrieve Row

Comments

Call Audit Enabled?

☐ Yes ☒ No

Call Audit Information

DIGITS ...

SQL Table Name

order ...

Column and Buffer Table

Column	Buffer
price	DATA EXCHANGE #1 ...
	DATA EXCHANGE #1 ...
	DATA EXCHANGE #1 ...
	DATA EXCHANGE #1 ...

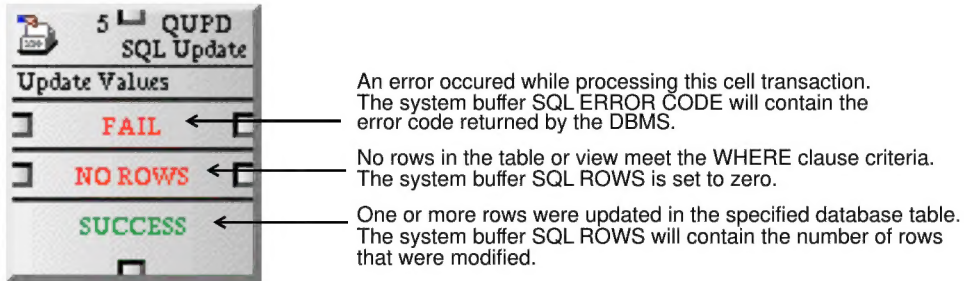
Where Clause

Logical	('s	Column	Type	Operator	Value	)'s
	(	order	Numeric ☐	>	1	)
And	(	status	String ☐	=	pending	)
			String ☐		DATA EXCHANGE #1	
			String ☐		DATA EXCHANGE #1	

Updating a row QUPD (SQL updated)

Use a QUPD cell to modify values in existing rows in a database table. QUPD parameters identify the table to be accessed. The rows to be modified are specified by a WHERE clause. Figure 2-12 shows a QUPD cell with the possible next cells. Figure 2-13 shows the QUPD parameters.

Figure 2-12
The QUPD cell



Entering an SQL statement in a QUPD cell

Suppose that you want to upgrade a customer's order priority in the "order" table to "highest" if the customer's order is older than 30 days. If the age of the order has been retrieved (by a previous QSEL cell) into buffer "age," you would code the following SQL statement to upgrade the order:

```
UPDATE order

SET priority = 'highest'

WHERE age > 30 ;
```

When you create the QUPD cell, you specify the following values in the cell parameter window as shown in Figure 2-13:

Screen Location	Field	Specify
	SQL Table Name	order
Under the Column and Value Table	Column	priority
	Value	highest
Under the Where Clause	Column	age
	Operator	>
	Value (	30

Figure 2-13
QUPD parameter window

QUPD Parameters

Cell #5 **QUPD SQL Update**

update values

Comments

Call Audit Enabled? Yes No

Call Audit Information DIGITS

SQL Table Name "order"

Column	Type	Value
priority	String	highest
	String	DATA EXCHANGE #1
	String	DATA EXCHANGE #1
	String	DATA EXCHANGE #1

More Less

Logical	('s	Column	Type	Operator	Value	)'s
		age	Numeric	>	30	
			String		DATA EXCHANGE #1	
			String		DATA EXCHANGE #1	
			String		DATA EXCHANGE #1	

Apply Cancel Help